How To Use MFC to Automate Excel and Fill a Range with an Array

This article was previously published under Q186120

Article ID : 186120
Last Review : July 13, 2004
Revision : 1.0

SUMMARY

This article demonstrates how to automate Microsoft Excel and fill a multi-cell range with an array of values.

MORE INFORMATION

To fill a multi-cell range without populating the cells one-by-one, you must create a two-dimensional variant SAFEARRAY which you pass to Excel by calling the SetValue function for the Range object. The following steps illustrate this process.

Notes for Automating Microsoft Excel 2000 and 2002

The sample code in this article uses class wrappers generated from the Excel 97 object library (Excel 8.olb). With slight modification, this code can be applied to an Automation client that uses class wrappers for Excel 2000 (Excel9.olb) or Excel 2002 (Excel.olb). For additional information about using the sample code described in this article with the Microsoft Excel 2000 or 2002 type library, please click the article number below to view it in the Microsoft Knowledge Base:

> 224925 (http://www.support.microsoft.com/kb/224925/EN-US/) INFO: Type Libraries for Office May Change With New Release

Steps to Create Project

1. Follow steps 1 through 12 in the following article in the Microsoft Knowledge Base to create a sample project that uses the IDispatch interfaces and member functions defined in the Excel8.olb type library:

   > 178749 (http://www.support.microsoft.com/kb/178749/EN-US/) How To Create an Automation Project Using MFC and a Type Library

2. To the dialog box created in steps 4 and 5 of the parent article 178749 (http://www.support.microsoft.com/kb/178749/EN-US/), add the following controls with properties as specified. Als o add the corresponding member variables:

Control	Name	Member Variable Type	Member Variable Name
Edit	IDC_STARTINGCELL	m_sStartingCell	CString
Edit	IDC_NUMROWS	m_iNumRows	short
Edit	IDC_NUMCOLS	m_iNumCols	short
CheckBox	IDC_STRING	m_bFillWithStrings	BOOL

3. At the top of the AutoProjectDlg.cpp file, add the following line:

   ```
   #include "excel8.h"
   ```

4. Add the following code to CAutoProjectDlg::OnRun() in the AutoProjectDlg.cpp file.

Sample Code

```
// OLE Variant for Optional.
COleVariant VOptional((long)DISP_E_PARAMNOTFOUND, VT_ERROR);

_Application objApp;

_Workbook objBook;
Workbooks objBooks;
Worksheets objSheets;
_Worksheet objSheet;
Range range;

if(!UpdateData(TRUE))
```

```
{
   return;
}

// Instantiate Excel and start a new workbook.
objApp.CreateDispatch("Excel.Application");
objBooks = objApp.GetWorkbooks();
objBook = objBooks.Add(VOptional);
objSheets = objBook.GetWorksheets();
objSheet = objSheets.GetItem(COleVariant((short)1));

//Get the range where the starting cell has the address
//m_sStartingCell and it's dimensions are m_iNumRows x m_iNumCols.
range = objSheet.GetRange(COleVariant(m_sStartingCell),
                          COleVariant(m_sStartingCell));
range = range.GetResize(COleVariant(m_iNumRows),
                        COleVariant(m_iNumCols));

//*** Fill the range with an array of values.

//Create the SAFEARRAY.
COleSafeArray saRet;
DWORD numElements[2];
numElements[0]= m_iNumRows;   //Number of rows in the range.
numElements[1]= m_iNumCols;   //Number of columns in the range.

if(m_bFillWithStrings)
{
   saRet.Create(VT_BSTR, 2, numElements);
}
else
{
   saRet.Create(VT_R8, 2, numElements);
}

//Fill the SAFEARRAY.
long index[2];
long iRow;
long iCol;

for(iRow=0;iRow<=m_iNumRows-1;iRow++)
{
   for(iCol=0;iCol<=m_iNumCols-1;iCol++)
   {
      index[0] = iRow;
      index[1] = iCol;
      if(m_bFillWithStrings)      //Fill with Strings.
      {
         VARIANT v;
         CString s;
         VariantInit(&v);
         v.vt = VT_BSTR;
         s.Format("r%dc%d", iRow, iCol);
         v.bstrVal = s.AllocSysString();
         saRet.PutElement(index, v.bstrVal);
         SysFreeString(v.bstrVal);
         VariantClear(&v);
      }
      else                     //Fill with Numbers.
      {
         double d;
         d = (iRow*1000) + iCol;
         saRet.PutElement(index, &d);
      }
   }
}

//Set the range value to the SAFEARRAY.
```

```
range.SetValue(COleVariant(saRet));
saRet.Detach();

//Return control of Excel to the user.
objApp.SetVisible(TRUE);
objApp.SetUserControl(TRUE);
```

5. Compile and Run the project.
6. Specify the following values for the controls on the dialog box:

Control	Contents
IDC_STARTINGCELL	A1
IDC_NUMROWS	10
IDC_NUMCOLS	5
IDC_STRING	True

   Click OK.

   Results: A new workbook is generated and cells A1:E10 of the first worksheet are populated with string values.
7. Specify the following values for the controls on the dialog box:

Control	Contents
IDC_STARTINGCELL	C3
IDC_NUMROWS	2
IDC_NUMCOLS	9
IDC_STRING	False

   Click OK.

   Results: A new workbook is generated and cells C3:K4 of the first worksheet are populated with numeric values.

REFERENCES

For more information about automating Microsoft Excel using MFC, please see the following articles in the Microsoft Knowledge Base:

186122 (http://www.support.microsoft.com/kb/186122/EN-US/) How To Use MFC to Automate Excel and Obtain an Array from a Range

184663 (http://www.support.microsoft.com/kb/184663/EN-US/) How To Embed and Automate a Microsoft Excel Worksheet with MFC

179706 (http://www.support.microsoft.com/kb/179706/EN-US/) Use MFC to Automate Excel and Create/Format a New Workbook

178781 (http://www.support.microsoft.com/kb/178781/EN-US/) How To Automate Excel Using MFC and Worksheet Functions

178783 (http://www.support.microsoft.com/kb/178783/EN-US/) How To Use MFC to Create a Microsoft Excel Chart

APPLIES TO

- Microsoft Excel 2000 Standard Edition
- Microsoft Visual C++ 5.0 Enterprise Edition

- Microsoft Visual C++ 6.0 Enterprise Edition
- Microsoft Visual C++ 5.0 Professional Edition
- Microsoft Visual C++ 6.0 Professional Edition
- Microsoft Visual C++ 6.0 Standard Edition
- Microsoft Excel 2002 Standard Edition
- Microsoft Excel 97 Standard Edition

Keywords: kbhowto kbinterop kbautomation kbcode KB186120